Amendments to the Specification:

Please replace the paragraph bridging pages 18-19 of the application, from page 18, line 4 through page 19, line 13, with the following:

> **Figure 5A** is a flowchart of processing performed by a graphing application for determining nominal vertical displacement of edges of a cyclic compound directed graph, such as graph **400** shown in **Figure 4**, in accordance with a preferred embodiment of the present invention. The graphing application is initiated (step **502**) by supplying a graph data structure as input to the graphing application. A node ordering subroutine of the graphing application is then invoked (step **504**) and the input graph is submitted to the node ordering subroutine (step **506**). The node ordering subroutine then evaluates the graph elements as described more fully below with reference to **Figure 5B**. An evaluation of the graph is made to determine if a recursive evaluation is required to identify any graph edges for inversion (step **510**). In accordance with a preferred embodiment of the invention, the evaluation of step **510** is performed by a graph evaluation subroutine described more fully below with reference to **Figure 5C**. If a recursive evaluation of the graph is not required, the graphing application generates a complete node order set from the results of the graph node evaluation of step **508** (step **518**). If a recursive evaluation of the graph is required, the graph is divided into partitions (step **512**). Division of the graph into graph partitions may be performed by a partitioning subroutine described more fully below with reference to **Figure 5D**. The node order of each graph partition is then solved by the node ordering subroutine (step **514**). The node order results of the graph partitions are then concatenated (step **516**). Concatenation of partition node order results may be performed by a concatenation subroutine described more fully below with reference to **Figure 5E**. A complete node order set defining a node order of the graph input to the graphing application at step **502** is then generated (step **518**) and one or more edges that may be inverted to minimize the number of inverted graph edges are identified (step ~~**518**~~ **520**). The graphing application then exits (step ~~**520**~~ **522**).